CHAT-PUB, BIWEEKLY_6TH

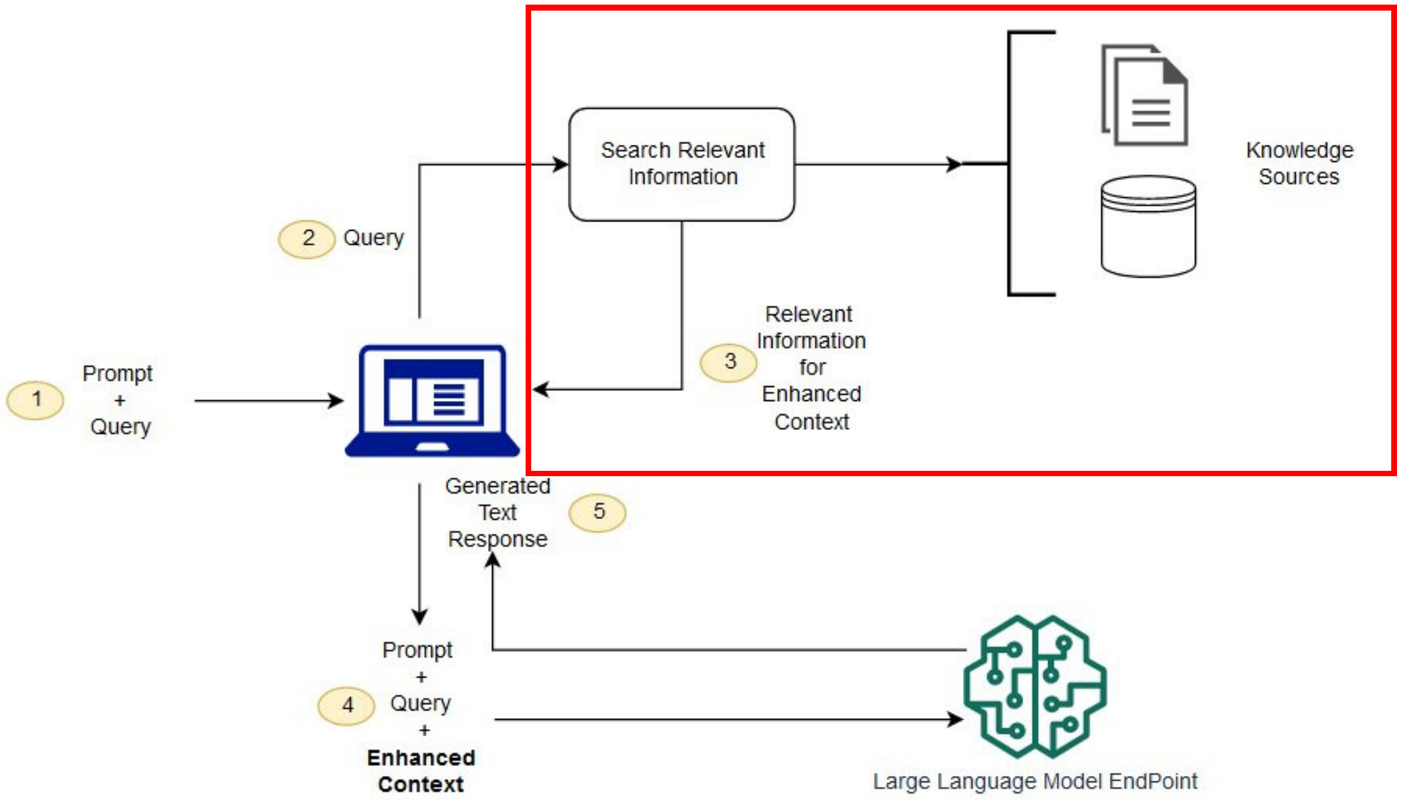
Helper of welfare policy for youth

김강산
양승빈
박진호
전창민

CONTENTS

TECHINAL BACKGROUND

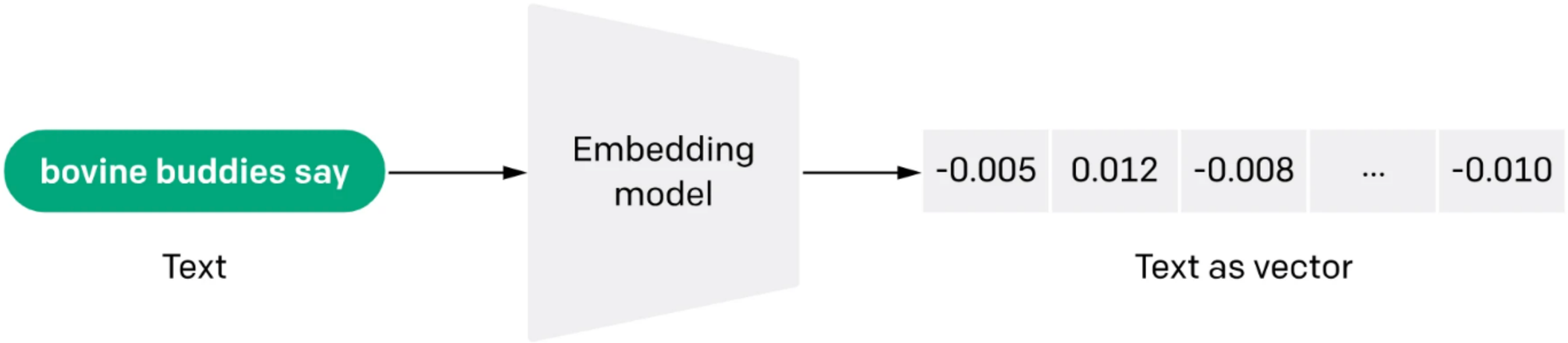
RETRIEVAL AUGMENTED GENERATION



HTTPS://DOCS.AWS.AMAZON.COM/SAGEMAKER/LATEST/DG/JUMPSTART-FOUNDATION-MODELS-CUSTOMIZE-RAG.HTML

TECHINAL BACKGROUND

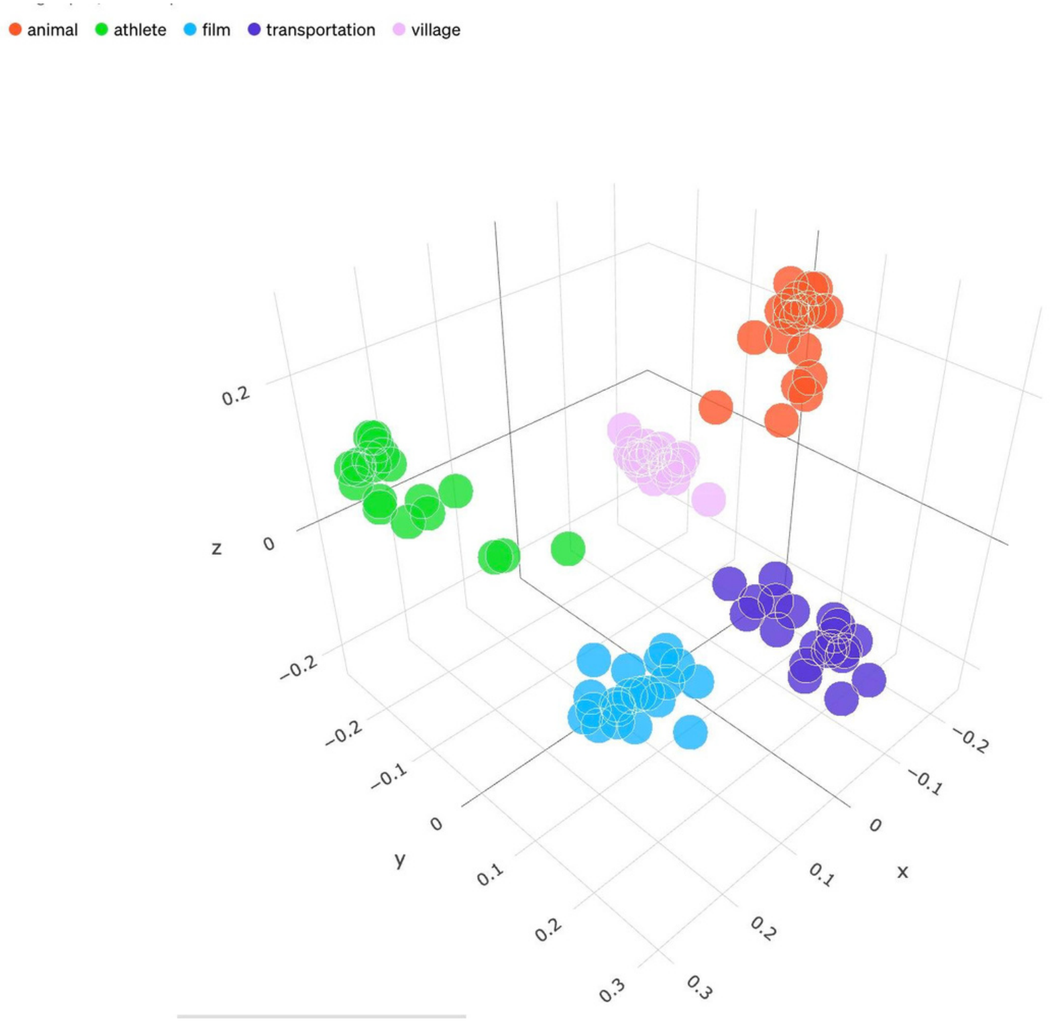
TEXT EMBEDDING



HTTPS://DIFY.AI/BLOG/TEXT-EMBEDDING-BASIC-CONCEPTS-AND-IMPLEMENTATION-PRINCIPLES

TECHINAL BACKGROUND

TEXT EMBEDDING

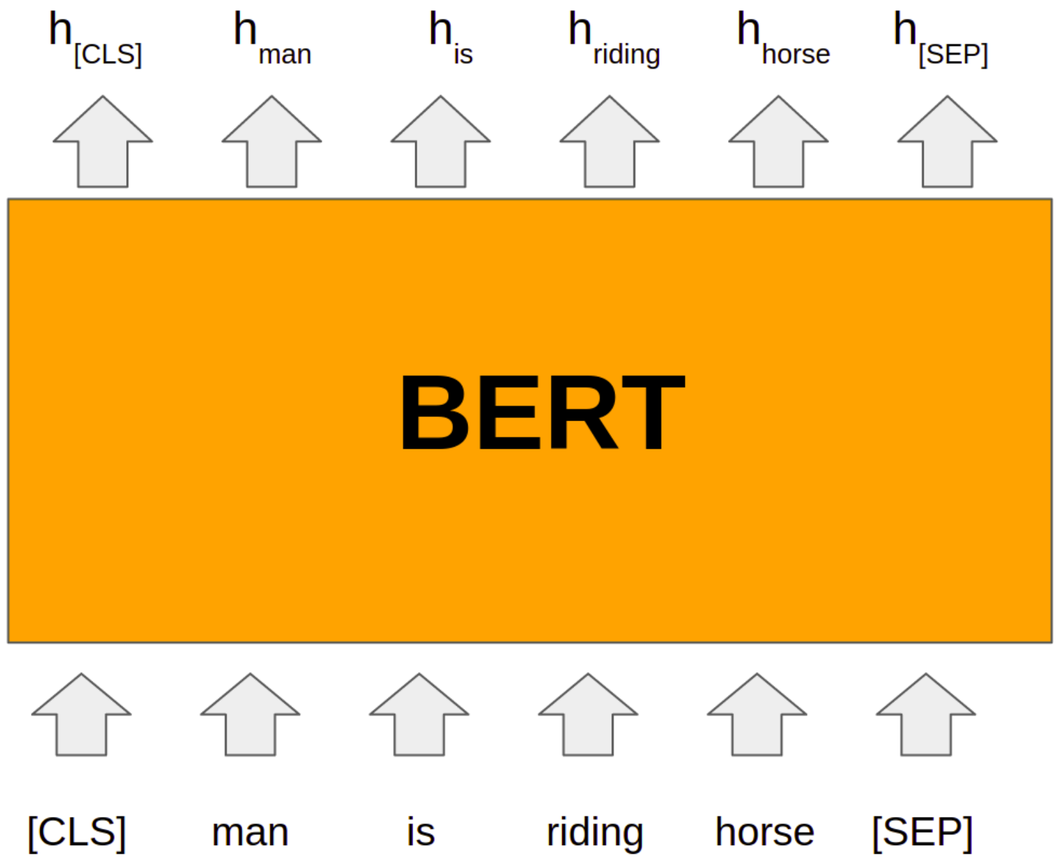


- Texts with similar meanings are embedded as vectors nearby.
- Texts with different meanings are embedded as vectors far away.
- We can use BERT-based models to perform text embedding.

HTTPS://DIFY.AI/BLOG/TEXT-EMBEDDING-BASIC-CONCEPTS-AND-IMPLEMENTATION-PRINCIPLES

TECHINAL BACKGROUND

BERT

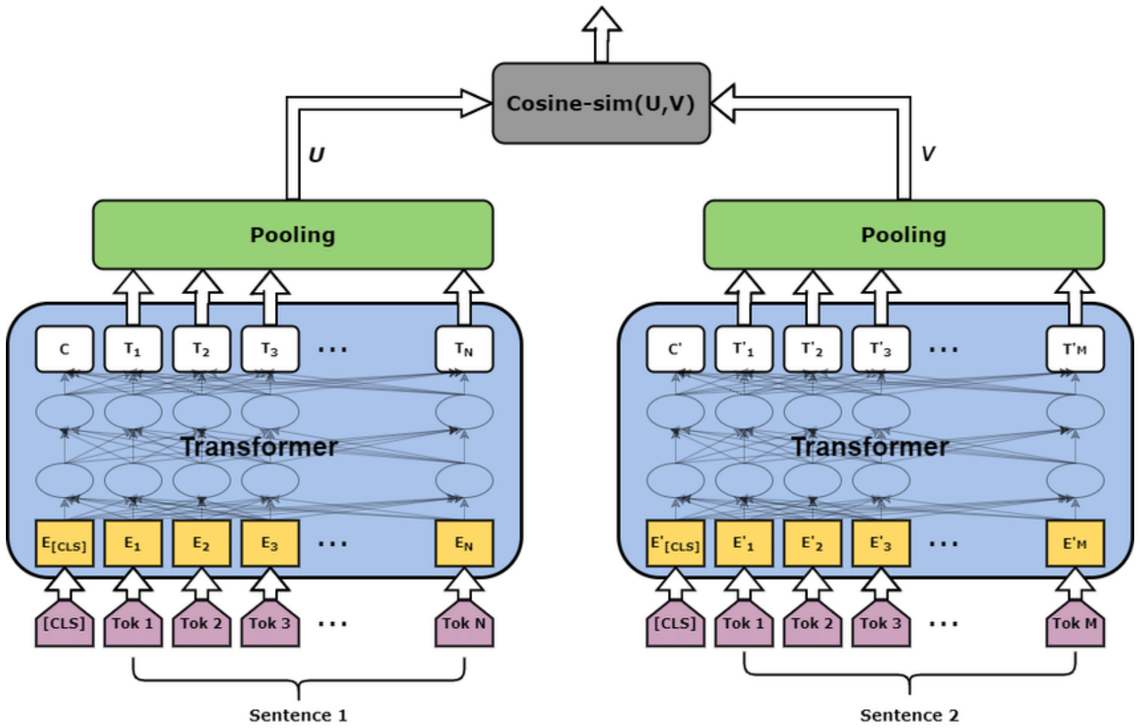


- BERT is a model that adopts the Encoder structure of the Transformer.
- Create embeddings for text utilizing contextual information

HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2021/05/ALL-YOU-NEED-TO-KNOW-ABOUT-BERT/

TECHINAL BACKGROUND

SENTENCE TRANSFORMER

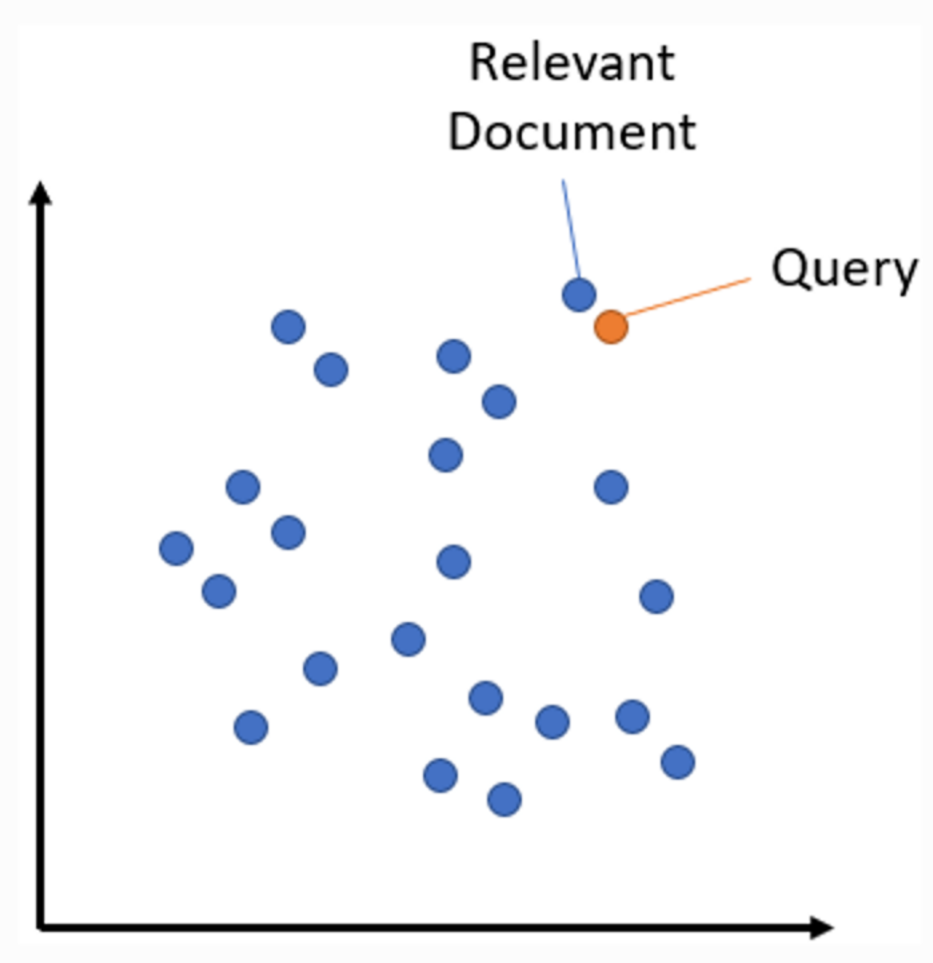


- SentenceTransformers is a Python framework for state-of-the-art sentence, text, and image embeddings.
- After text embedding is performed using the Sentence Transformers (BERT structure), similarity can be measured by cosine similarity.

HTTPS://WWW.RESEARCHGATE.NET/FIGURE/SIAMESE-SENTENCE-TRANSFORMER-STRANSFORMER-ARCHITECTURE_FIG2_353487642

TECHINAL BACKGROUND

EMBEDDING-BASED RETRIEVAL

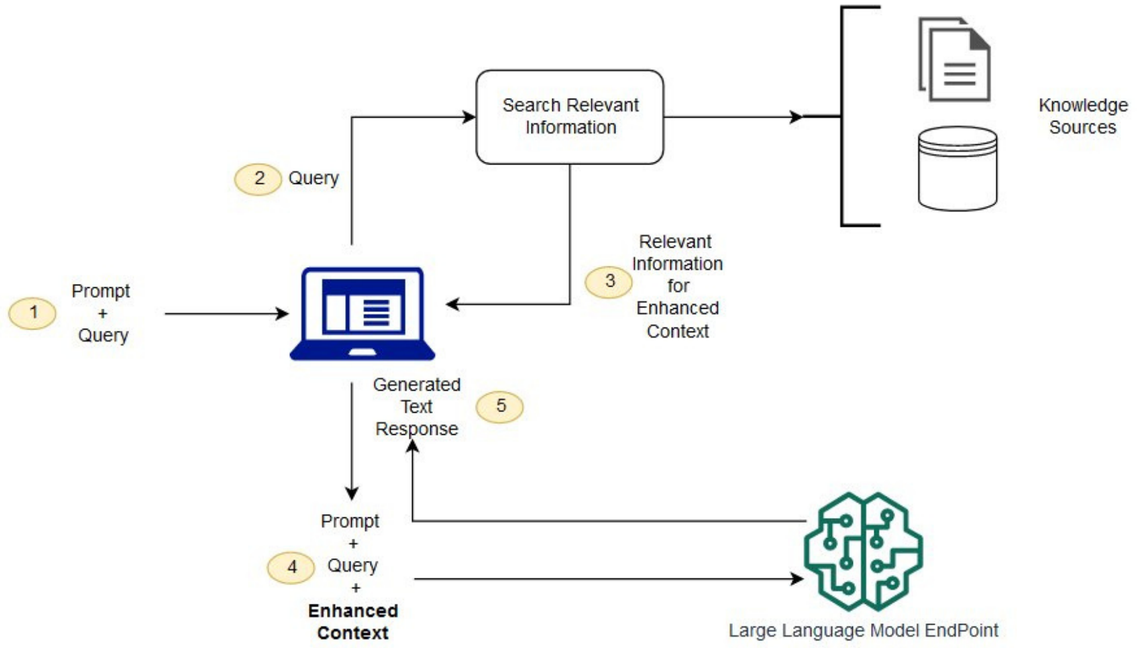


- When a user's question is asked, the embedding vector for the question and the embedding vector for the related document are located in a close space.
- Therefore, we can retrieve the document content needed to answer the question using embedding!

HTTPS://WWW.SBERT.NET/EXAMPLES/APPLICATIONS/INFORMATION-RETRIEVAL/README.HTML

TECHINAL BACKGROUND

RETRIEVAL AUGMENTED GENERATION



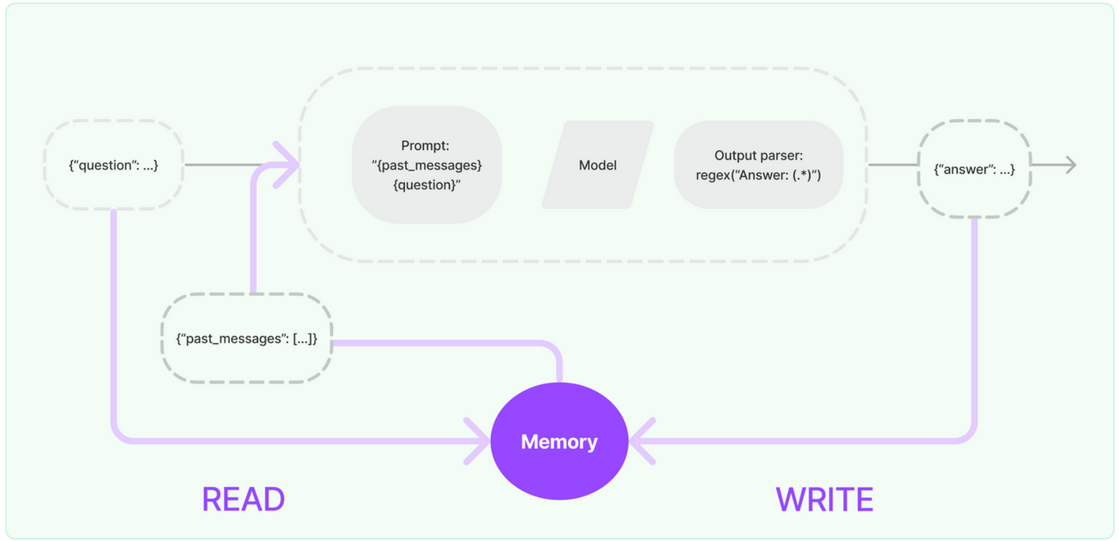
- Retrieval Augmented Generation (RAG) technology enables generative models to use external knowledge (not utilized for training) when performing answers.

- Perform a retrieval for a given user question to extract information from the knowledge store and combine it with a prompt to generate an answer.

HTTPS://DOCS.AWS.AMAZON.COM/SAGEMAKER/LATEST/DG/JUMPSTART-FOUNDATION-MODELS-CUSTOMIZE-RAG.HTML

TECHINAL BACKGROUND

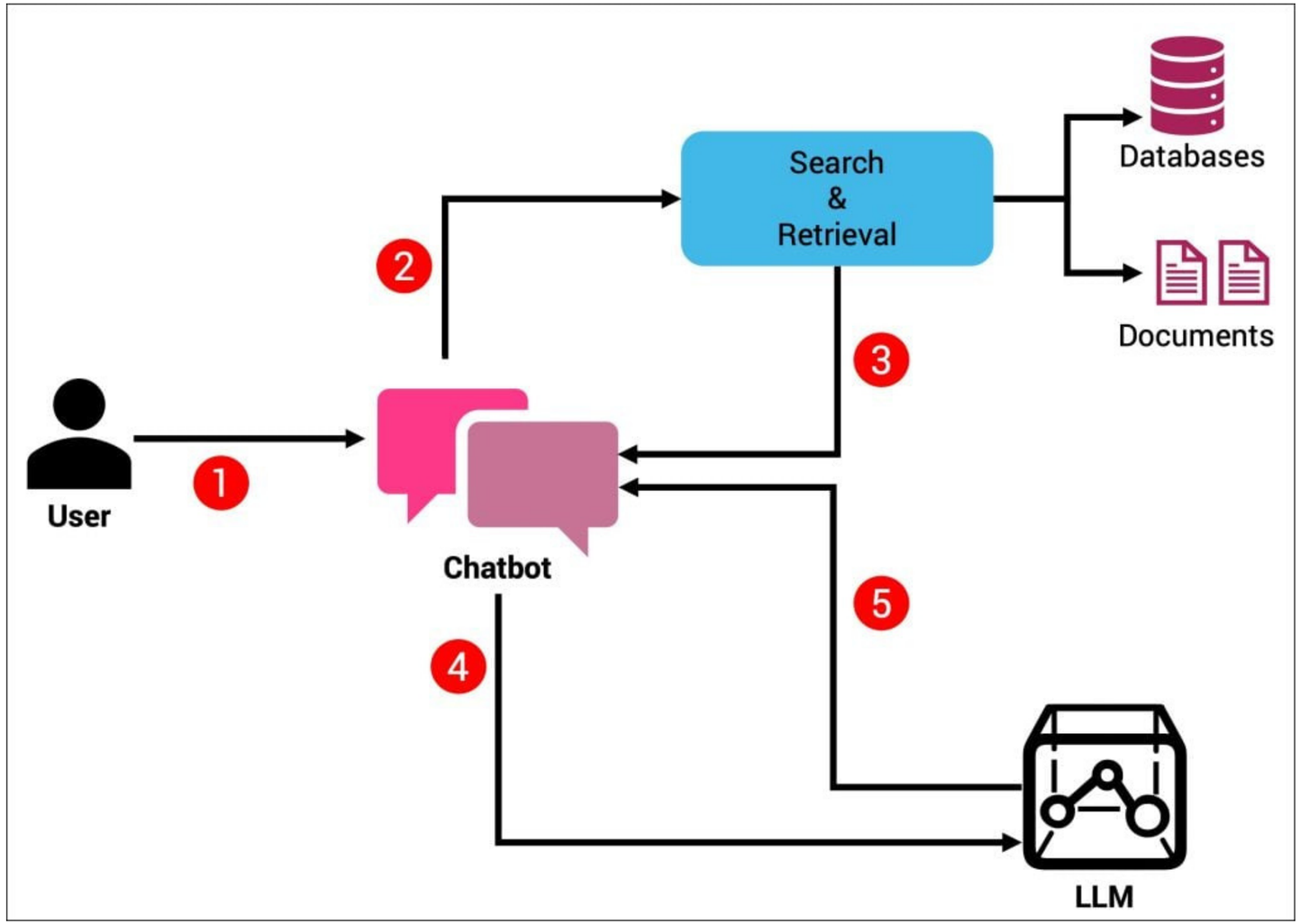
MEMORY



- The chatbot should remember the context of the previous conversation and refer to it in generating answers.
- We will utilize the memory capabilities provided by Langchain.
- Performs saving the user's speech and the chatbot's answers.
The saved conversation context is utilized to generate Retrieval and answers.ODO

HTTPS://PYTHON.LANGCHAIN.COM/DOCS/MODULES/MEMORY/

MODEL PIPELINE



MODEL PIPELINE

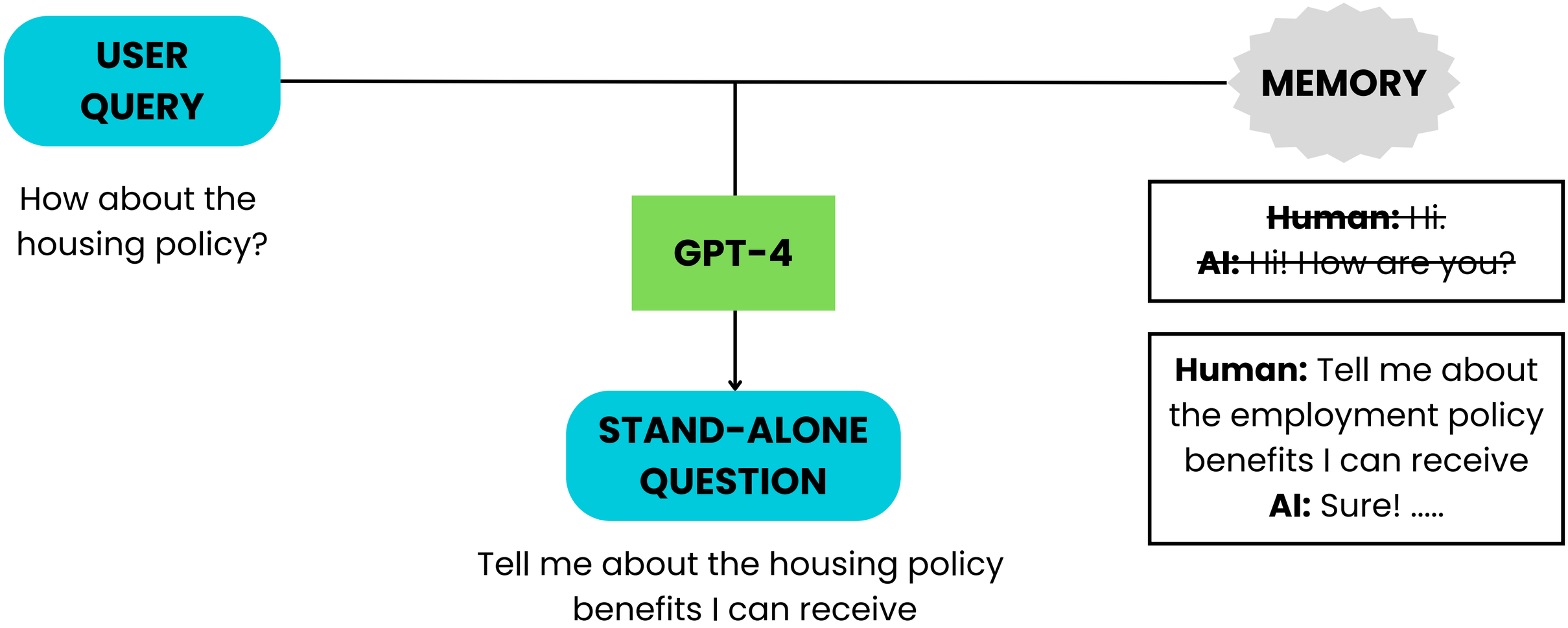
1. QUERY INPUT

USER QUERY

How about the housing policy?

MODEL PIPELINE

2. MAKE STAND-ALONE QUESTION



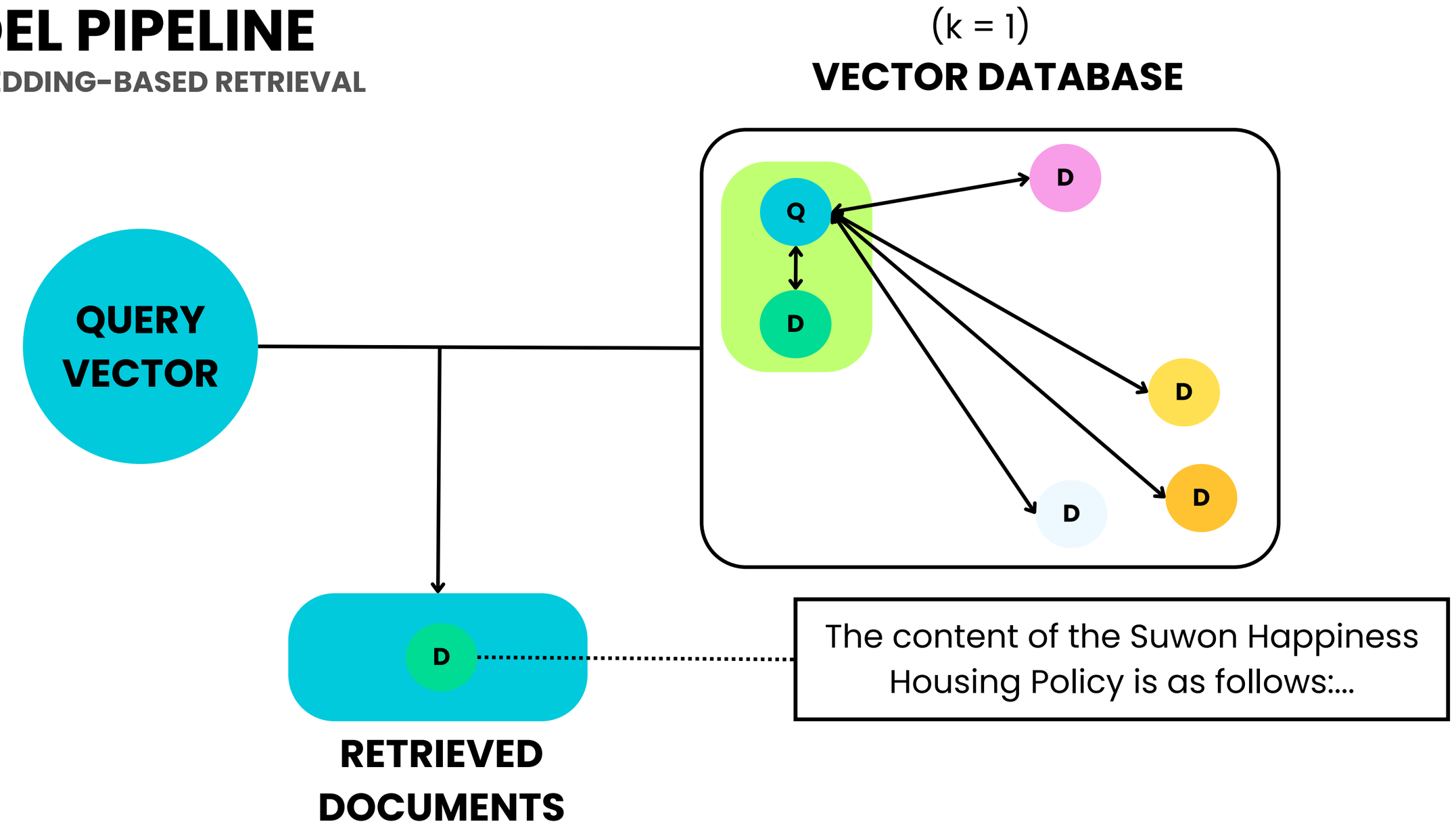
MODEL PIPELINE

3. MAKE QUERY VECTOR

Tell me about the housing policy benefits I can receive

MODEL PIPELINE

4. EMBEDDING-BASED RETRIEVAL



MODEL PIPELINE

5.CHAT COMPLETION

STAND-ALONE QUESTION

Tell me about the housing policy benefits I can receive

+

RETRIEVED DOCUMENTS

The content of the Suwon Happiness Housing Policy is as follows:...

GPT-4

RESPONSE

APPENDIX

PROMPT EXAMPLE – STAND ALONE QUESTION

Given the following conversation and a follow up question, rephrase the follow up question to be a standalone question.
If you do not know the answer reply with 'I am sorry'.

Chat History:

MEMORY

MEMORY BUFFER

Follow Up Input:

USER QUERY

APPENDIX

PROMPT EXAMPLE - CHAT COMPLETION

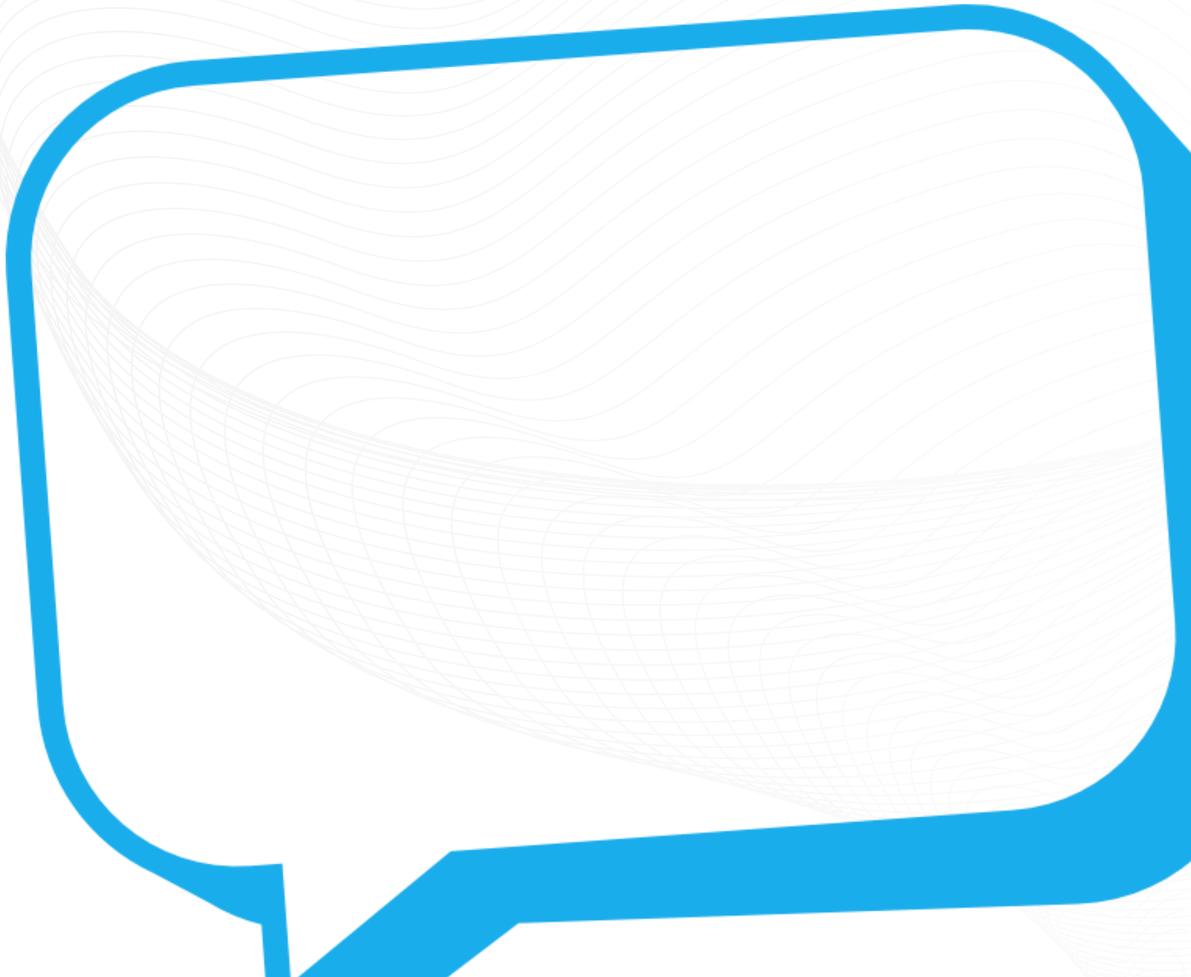
Answer the user's questions using the given context.
If you do not know the answer reply with 'I am sorry'.
Context:

RETRIEVED DOCUMENTS

Question:

STAND-ALONE QUESTION

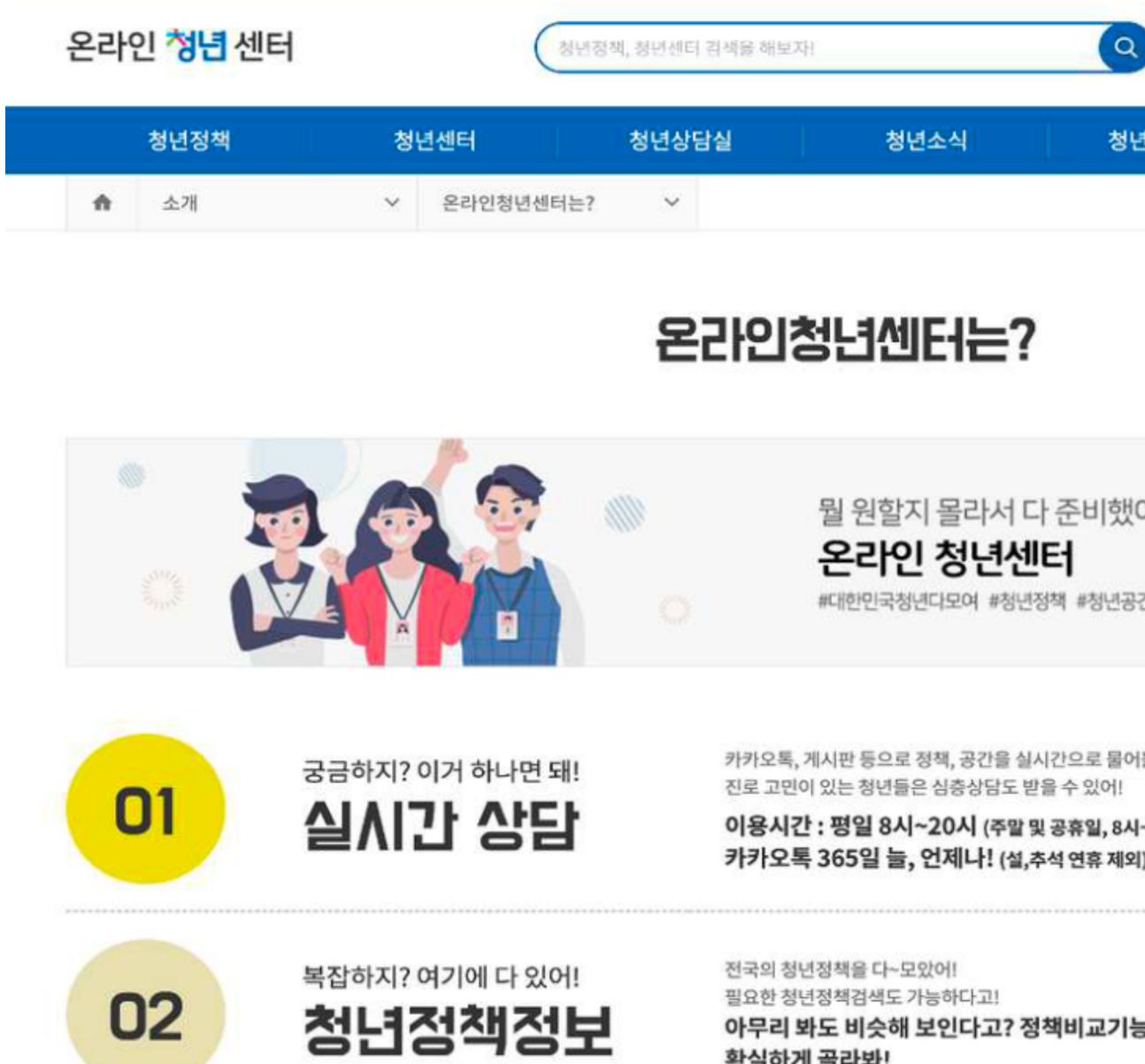
DATA CRAWLING



DATA CRAWLING

MAIN PROBLEMS...

- What DATA do we need?
- What scraping techniques or tools will be used?
- How will the scraped data be stored and structured?
- etc...



DATA CRAWLING

CODE PROCESS

get_id()

doc_parser()

result_list

```python
def get_id():…

def doc_parser(contents):…

if __name__ == '__main__':
    print("Start crawling...")
    YPlist = get_id()

    print(f"The number of data : {len(YPlist)}")

    result_list = list()

    for index, contents in enumerate(tqdm(YPlist)):
        print(index,contents)
        parsed_data = doc_parser(contents)
        result_list.append({
            'YP': index,
            'title': contents[0],
            'R-number': contents[1],
            'contents': parsed_data,
        })
```

DATA CRAWLING

DATA FIELD

Data Field of four main dictionary

1. summary dict

{
'정책 번호'
'정책 분야'
'지원 내용'
'사업 운영 기간'
'사업 신청 기간'
'지원 규모(명)'
'비고'
}

2. qualification dict

{
'연령'
'거주지 및 소득'
'학력'
'전공'
'취업 상태'
'특화 분야'
'추가 단서 사항'
'참여 제한 대상'
}

3. methods dict

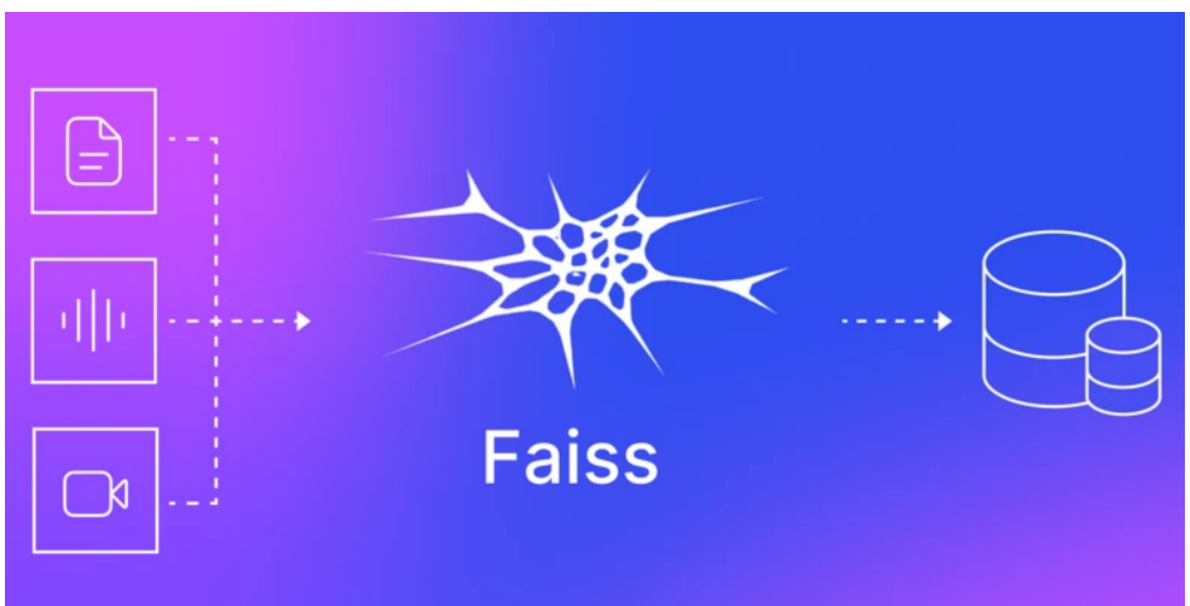
{
'신청 절차'
'심사 및 발표'
'신청 사이트'
'제출 서류'
}

4. etc dict

{
'기타 유익 정보'
'주관 기관'
'운영 기관'
'사업관련 참고 사이트 1'
'사업관련 참고 사이트 2'
'첨부파일'
}

DATA CRAWLING

DATA VECTORIZATION




```
search(model, index, '구리시에 관심이 있는데 괜찮은 정책 있어?', 1)
```

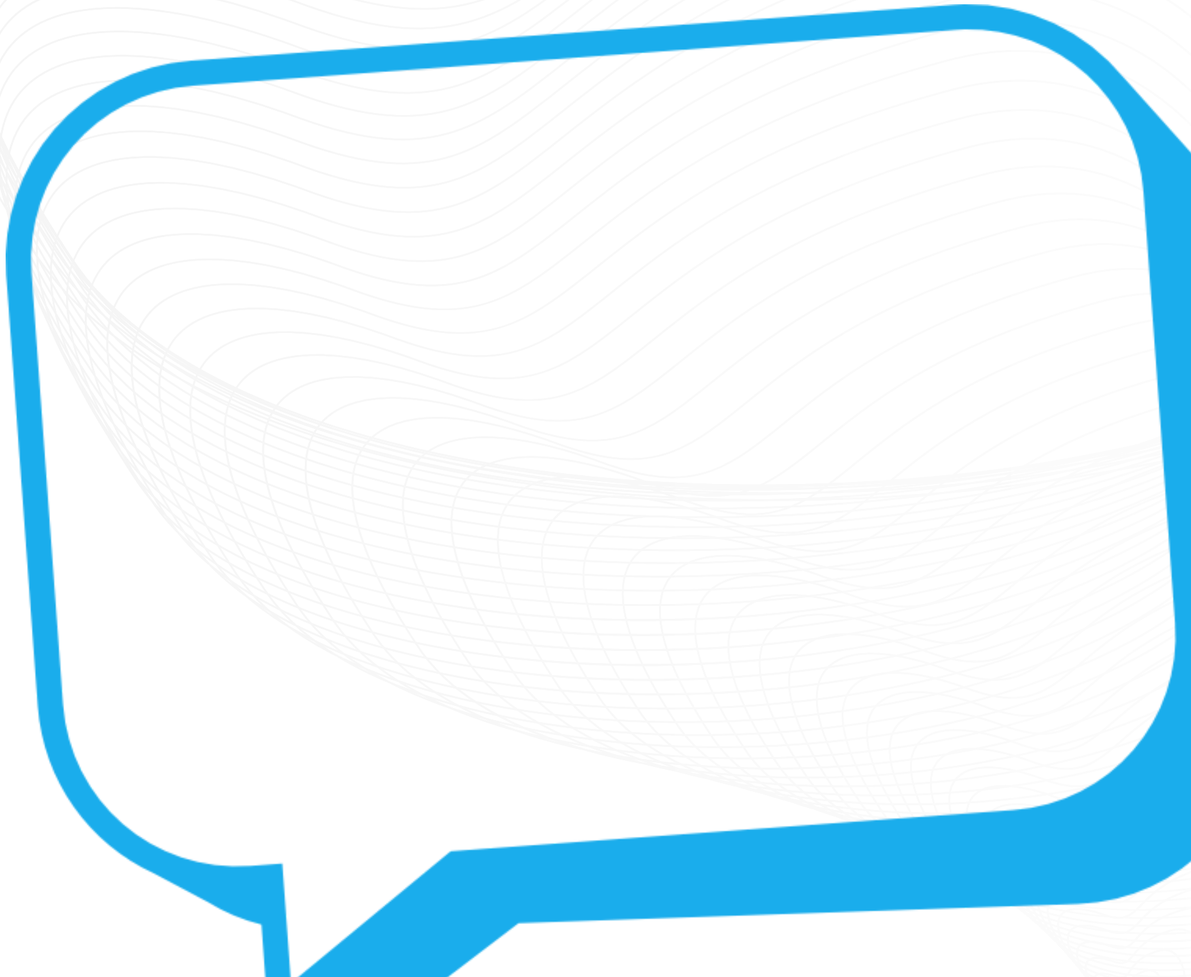
✓ 0.1s

```
[[ 2.22022831e-02 -5.36617100e-01 -8.72454569e-02 -6.93040669e-01
   1.95074722e-01 -1.01507910e-01  8.59122872e-01 -1.15090422e-01
   3.60790044e-01 -4.36832309e-02 -2.85417706e-01 -2.49668226e-01
   3.11255962e-01  2.12615371e-01 -1.44003153e-01 -4.72312391e-01
   2.29062319e-01 -7.68211007e-01  2.83244610e-01 -1.10166359e+00
  -1.32374123e-01  6.76300451e-02 -2.11506173e-01  3.73165399e-01
  -1.00836039e+00  6.11341655e-01  6.28062263e-02  5.42643726e-01
  -6.65344298e-01  4.01938856e-01  3.73282760e-01 -3.87211710e-01
  -1.28040284e-01 -6.10775352e-01 -1.07950374e-01  6.16771996e-01
  -3.65155518e-01 -6.71888828e-01  3.09403148e-02  2.73636281e-02
  -6.52105689e-01 -3.30289453e-01 -2.31157631e-01  3.83417867e-02
   4.56821054e-01 -8.11878219e-02 -1.60433561e-01 -5.22067487e-01
  -4.73977953e-01  3.92300069e-01  2.55464017e-01 -7.35937208e-02
   1.92174111e-02  5.84220409e-01  2.93138891e-01 -3.57280940e-01
   4.07842398e-01 -1.53997496e-01 -4.41726208e-01  2.25594983e-01
   4.25655574e-01  2.78181404e-01  6.44712523e-02  2.45962977e-01
   5.09047834e-03 -3.41961235e-01 -8.38873804e-01 -1.16708264e-01
  -8.86502802e-01  2.51761079e-02 -7.09160447e-01 -1.90470845e-01
   9.20764089e-01  4.97447550e-01  2.15500265e-01 -6.28849030e-01
  -1.88646719e-01  6.23598456e-01  1.08124125e+00 -8.47105756e-02
  -5.74157357e-01 -5.36367074e-02 -2.01641440e-01 -1.99187666e-01
  -4.24037836e-02  8.57522130e-01 -2.33404730e-02  2.58832037e-01
   2.82135308e-01 -4.09846485e-01  8.96249190e-02  4.12386889e-03
  -1.73793897e-01 -5.38320206e-02 -5.77122569e-01  1.66340634e-01
  -1.62463889e-01 -1.01045065e-01  1.23788230e-01  1.73807904e-01
...
summary: {'정책 번호': 'R2023060212999', '정책 분야': '교육분야', '지원 내용': '
qualification: {'연령': '만 18세 ~ 39세', '거주지 및 소득': '사업개시일 현재 만18
methods: {'신청 절차': '구비 서류 양식 작성 및 증빙서류 구비', '심사 및 발표': '○ 선
etc: {'기타 유의 정보': '-', '주관 기관': '구리시청 일자리경제과', '운영 기관': '구리
```

DATA CRAWLING

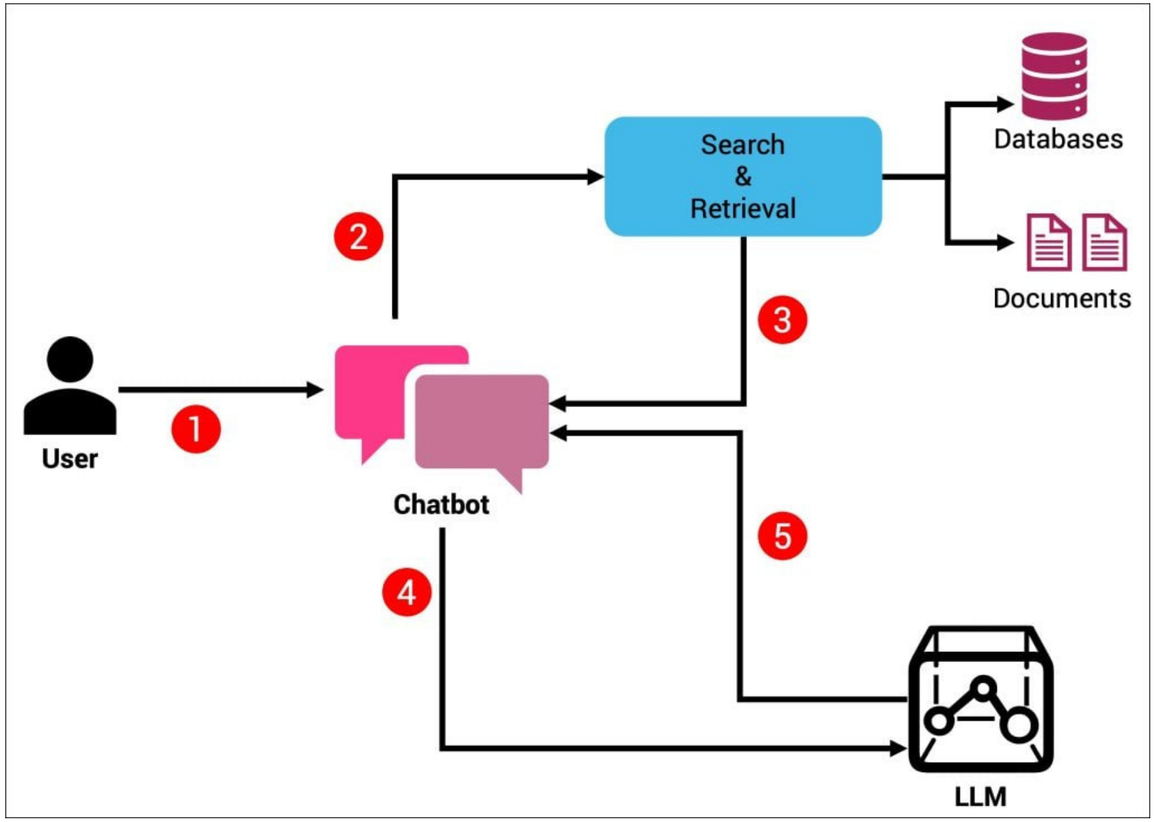
DATABASE

DEMO



DEMO

Streamlit



DEMO

PROJECT SCHEDULE

Task	week 3	week 4	week 5	week 6	week 7	week 8	week 9	week 10	week 11	week 12	week 13	week 14	week 15
Topic Study													
Data Collecting													
Modeling													
Front-End													
Back-End													
Testing													